Stream Programming Environments

Pat Hanrahan
Computer Science & Electrical Engineering
Stanford University

GP^2 Workshop
August 7-8, 2004

Acknowledgements

- Bill Dally
- Eric Darve
- Vijay Pande
- Bill Mark
- John Owens
- Kurt Akeley
- Mark Horowitz
- Ian Buck
- Mattan Erez
- Kayvon Fatahalian
- Tim Foley
- Daniel Horn
- Michael Houston
- Jeremy Sugarman

Funding: DARPA, DOE, ATI, IBM, NVIDIA, SONY
Motivation

- Cinematic games and media drive GPU market
- Current GPU faster than CPU (at graphics)
- Gap between the GPU and the CPU increasing
- Why? Data parallelism; efficient communication
- Programmable GPUs ≈ Stream processors
- Many applications map to stream processing
- Therefore, a $50 high-performance parallel-computer is shipping with every PC
- Revolutionize computing

Overview

- Technology trends
- Stream programming abstraction
- Brook for GPUs
- Applications
VLSI for Programmers :-)

3
The Capability Gap

Recent Performance Trends
Stream Abstraction

Streams – Old/Hot Idea in CS

- `<stream.h>`
- OpenGL/GLS/Chromium
- Data visualization systems (vtk, avs, dx)
- Signal processing (signal flow graphs)
- Functional programming
- Streaming data bases
- Sensor nets
Minimize State!

Fragment Processor

- SIMD Architecture
- Dual Issue / Co-Issue
- FP32 Computation
- Shader Model 3.0
Stream Programming Abstraction

- Streams
  - Collection of data records
- Kernels
  - Inputs/outputs are streams
  - Perform computation
  - Can be chained together
Why Architects like Streams?

- Parallelism
  - Data parallelism
  - Pipeline (task) parallelism
- Communication
  - Producer-consumer locality
  - Predictable memory access pattern
  - No read-write hazards; simple coherence
  - Hide latency of random memory accesses
  - High arithmetic intensity

* A lot like vector machines ...

---

Arithmetic Intensity

Arithmetic Intensity = Compute-to-bandwidth ratio

Graphics pipeline
- Vextex
  BW: 1 vertex = 32 bytes;
  OP: 100-500 f32-ops / vertex
- Fragment
  BW: 1 fragment = 10 bytes
  OP: 300-1000 i8-ops/fragment
Measured Arithmetic Intensity

Microbenchmarks

<table>
<thead>
<tr>
<th></th>
<th>GFLOPS</th>
<th>Cache BW</th>
<th>Seq BW</th>
</tr>
</thead>
<tbody>
<tr>
<td>NV 5900 Ultra</td>
<td>40.0</td>
<td>11.4</td>
<td>4.1</td>
</tr>
<tr>
<td>NV 6800 Ultra</td>
<td>53.4</td>
<td>20.6</td>
<td>8.4</td>
</tr>
<tr>
<td>ATI 9800 XT</td>
<td>26.1</td>
<td>12.2</td>
<td>7.3</td>
</tr>
<tr>
<td>ATI X800 XT PE*</td>
<td>63.7</td>
<td>28.4</td>
<td>15.6</td>
</tr>
</tbody>
</table>

Bandwidth measured in GB/sec.

* ATI X800 XT PE is a prerelease board: 500Mhz core / 500Mhz clock

GPUBench: Evaluating GPU performance for numerical and scientific applications, K. Fatahalian, I. Buck, M. Houston, P. Hanrahan, GP^2 2004

CPU vs GPU

- Intel 3 Ghz Pentium 4
  - 12 GFLOPS peak performance (via SSE2)
  - 6 GB/sec peak memory bandwidth
  - 44 GB/sec peak bw from 8K L1 data cache
- NVIDIA GeForce 6800
  - 45 GFLOPS peak performance
  - 36 GB/sec peak memory bandwidth
  - 21 GB/sec peak bw from ?K L1 data cache
Approach I

Map application to graphics primitives
- Graphics library-based programming models
  - Cg/HLSL
  - OpenGL Shading Language
  - Sh [McCool et al. 2004]

Approach II

Map application to parallel computer
- Stream languages
  - AWK, Ptolemy, ...
  - StreaMIT, StreamC/KernelC, ...
- Data parallel programming
  - APL, SETL, S, Fortran90, ...
  - C* (lisp*), NESL, ...
- Communicating sequential processes (CSP)
  - Threads: Occam, UPC
  - Message passing: MPI
Stream Programming Environment

- Collections stored in memory
  - Multidimensional arrays (stencils)
  - Graphs and meshes (topology)
- Data parallel operators
  - Application: map
  - Reductions: scan, reduce (fold)
  - Communication: send, sort, gather, scatter
  - Filter (|O|<|I|) and generate (|O|>|I|)

Brook

Ian Buck, Ph. D. Thesis, Stanford

Brook for GPUs: Stream computing on graphics hardware,
I. Buck, T. Foley, D. Horn, J. Sugarman, K. Fatahalian,
M. Houston, P. Hanrahan, SIGGRAPH 2004
Brook Language

```c
kernel void foo (  
    float a<>, float b<>,  
    out float result<>  )  
{
    result = a + b;
}
```

```c
float a<100>;
float b<100>;
float c<100>;
foo(a,b,c);  
```

```c
for (i=0; i<100; i++)  
c[i] = a[i]+b[i];
```

Goals

- Develop version of PCA Brook for GPUs
  - Programmer need not know GL
- Versions
  - New ATI (420) and NVIDIA (NV40) hardware
  - Linux and Windows
  - DX and OpenGL
- Release as open source [V1.0 Dec 2003]
  - http://sourceforge.net/projects/brook
  - over 6,300 downloads in 8 months
Brook Performance

First Generation GPUs

Floating precisions different
- ATI – 24-bit
- NV – ~IEEE 32-bit
- Intel – IEEE 32-bit

compared against:
- Intel Math Library
- Atlas Math Library
- Cached blocked segmentation
- FFTW
- SSE-opt Ray Triangle code

Second Generation GPUs

Floating precisions different
- ATI – 24-bit
- NV – ~IEEE 32-bit
- Intel – IEEE 32-bit

compared against:
- Intel Math Library
- Atlas Math Library
- Cached blocked segmentation
- FFTW
- SSE-opt Ray Triangle code
Matrix-matrix multiplication is bandwidth limited on GPU.
- Memory blocking to increase cache utilization does not help
- Architectural problem, not programming model problem

* ATI X800 XT PE is a prerelease board: 500Mhz core/500Mhz clock

Beyond Graphics and Imaging ...

Accelerating molecular dynamics with GPUs, I. Buck, V. Rangasayee, E. Darve, V. Pande, P. Hanrahan GP^2 2004

Applications

- Media: audio, images (vision), video, ...
- Simulation
  - Monte Carlo
    - Ray tracing
  - Ordinary differential equations
    - N-body problems: molecular dynamics, astrophysics
    - Particle systems and rigid body dynamics
  - Partial differential equations
    - Explicit: elastic deformations
    - Implicit: cloth, fluid flow
- Machine learning and computational statistics?
16 Node GPU Cluster

- **Compute**
  - 32 2.4GHz P4 Xeons
  - 16GB DDR
  - 1.2TB disk
  - Intel E7505 chipset

- **Network**
  - Infiniband 4X interconnect
  - GigE

- **Graphics**
  - ATI Radeon 9800 Pro 256MB

Parallel computation on a cluster of GPUs, M. Houston, K. Fatahalian, J. Sugarman, I. Buck, P. Hanrahan GP^2 2004

Wrap-Up

Vision

- Cinematic games and media drive GPU market
- Current GPU faster than CPU (at graphics)
- Gap between the GPU and the CPU increasing
- Why? Data parallelism; efficient communication
- Programmable GPUs = Stream processors
- Many applications map to stream processing
- Therefore, a $50 high-performance parallel-computer is shipping with every PC
- Revolutionize computing
Opportunities

• Current hardware not optimal
  – Incredible opportunity for architectural innovation

• Current software environment immature
  – Incredible opportunity for reinventing parallel computing software

Questions?

Fly-fishing fly images from The English Fly Fishing Shop